**TRIANGLE PROBLEM**

def is\_triangle(a, b, c):

return 1 <= a <= 20 and 1 <= b <= 20 and 1 <= c <= 20

def triangle\_type(a, b, c):

if a == b == c:

return "Equilateral triangle"

elif a != b and a != c and b != c:

return "Scalene triangle"

else:

return "Isosceles triangle"

while True:

try:

a, b, c = map(int, input("\nEnter 3 integers which are sides of a triangle: ").split())

except ValueError:

print("Invalid input. Please enter integers.")

continue

if is\_triangle(a, b, c):

print(triangle\_type(a, b, c))

break

else:

print("Triangle sides should be integers between 1 and 20. Please try again.")

**COMMISION PROBELM**

def calculate\_commission(sales):

if sales > 1800:

return 0.10 \* 1000 + 0.15 \* 800 + 0.20 \* (sales - 1800)

elif sales > 1000:

return 0.10 \* 1000 + 0.15 \* (sales - 1000)

else:

return 0.10 \* sales

locks\_price, stocks\_price, barrels\_price = 45.0, 30.0, 25.0

total\_locks, total\_stocks, total\_barrels = 0, 0, 0

while True:

try:

locks = int(input("\nEnter the number of locks (Enter -1 to exit): "))

except ValueError:

print("Invalid input. Please enter an integer.")

continue

if locks == -1:

break

stocks = int(input("Enter the number of stocks: "))

barrels = int(input("Enter the number of barrels: "))

if not 1 <= locks <= 70 or not 1 <= stocks <= 80 or not 1 <= barrels <= 90:

print("Invalid input. Values should be within the range.")

continue

total\_locks += locks

total\_stocks += stocks

total\_barrels += barrels

print(f"Total locks: {total\_locks}\nTotal stocks: {total\_stocks}\nTotal barrels: {total\_barrels}")

total\_sales = locks\_price \* total\_locks + stocks\_price \* total\_stocks + barrels\_price \* total\_barrels

print(f"\nTotal sales: {total\_sales}")

if total\_sales > 0:

commission = calculate\_commission(total\_sales)

print(f"The commission is: {commission}")

else:

print("There are no sales")

**NEXT DATE**

def is\_leap(year):

return year % 400 == 0 or (year % 100 != 0 and year % 4 == 0)

def get\_next\_date(day, month, year):

days\_in\_month = [0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]

if month == 2:

max\_days = 29 if is\_leap(year) else 28

elif month in [1, 3, 5, 7, 8, 10, 12]:

max\_days = 31

else:

max\_days = 30

if day < 1 or day > max\_days:

print(f"Invalid date: {month}/{day}/{year} has an invalid day.")

return None

day += 1

if day > max\_days:

day = 1

month += 1

if month > 12:

month = 1

year += 1

return day, month, year

def main():

day = int(input("Enter the day: "))

month = int(input("Enter the month: "))

year = int(input("Enter the year: "))

next\_date = get\_next\_date(day, month, year)

if next\_date is not None:

tomm\_day, tomm\_month, tomm\_year = next\_date

print(f"The given date: {day} {month} {year}")

print(f"The next date: {tomm\_day} {tomm\_month} {tomm\_year}")

if \_name\_ == "\_main\_":

main()